サポートベクターマシンによる指動作識別と積分筋電位を利用した筋電義手制御に関する研究

著者	菊地 毅
著者別名	KIKUCHI Takeshi
その他のタイトル	IDENTIFICATION OF FINGER OPERATION USING SUPPORT VECTOR MACHINE AND CONTROL OF MYOELECTRIC PROSTHETIC HAND BASED ON INTEGRATED ELECTROMYOGRAM
ページ	1-40
発行年	2015-03-24
学位授与年月日	2015-03-24
学位名	修士(工学)
学位授与機関	法政大学 (Hosei University)
URL	http://hdl.handle.net/10114/11735

2014 年度　修士論文

論文題名

サポートベクターマシンによる指動作識別と

積分筋電位を利用した筋電義手制御に関する研究

IDENTIFICATION OF FINGER OPERATION USING SUPPORT VECTOR MACHINE AND CONTROL OF MYOELECTRIC PROSTHETIC HAND BASED ON INTEGRATED ELECTROMYOGRAM

指導教員　石井　千春　教授

法政大学大学院　理工学研究科

機械工学専攻　修士課程

学生証番号　　13R1110

氏名　　菊地（キクチ）　毅（タケシ）

2014 年度　修士論文

論文題目 サポートベクターマシンによる指動作識別と

積分筋電位を利用した筋電義手制御に関する研究

ふりがな
氏名　菊地（きくち）　毅（たけし）

研究科専攻　理工学研究科機械工学専攻

学籍番号　13R1110

指導教員　石　井　千　春

修了年月(西暦)　2015 年　3 月

法政大学大学院

Abstract

Myoelectric prosthetic hands, which use surface electromyograms (SEMG) to identify the intended motion and the control movement of the artificial hand accordingly, have been studied for many years. Various signal processing and identification methods have greatly expanded the possibilities for studying myoelectric hands and many recent studies have been practical, with a focus on commercialization.

In this study, the identification of finger operation and control of the fingers of a myoelectric prosthetic hand are discussed. An integrated electromyogram (IEMG) is used to extract features from finger operations, and a support vector machine-based classifier (SVM) is employed to identify the six different types of finger operation. To enhance the myoelectric prosthetic hand operability, two new control methods are proposed. In the first method, the prosthetic finger angle is changed based on the duration of the muscular activity during finger operation. The strength of the muscular activity is not reflected. Thus, the target angle of the finger is determined without considering the degree of the muscular activity. In the second control method, the prosthetic finger angle is changed based on not only the duration but also the degree of the muscular activity of the finger during finger operation. A contribution of this paper is that the formula to give a target angle of the finger in which the identification result of finger operation, the duration of the muscular activity and the degree of the muscular activity are reflected, was established.

As each finger is controlled based on the degree of muscular activity, the second method enables users to adjust the prosthetic hand's grasp on objects using their intent. Therefore, this method facilitates the grasping of complex-shaped objects. Although some commercial myoelectric prosthetic hands can adjust finger speed in proportion to the strength of muscular activity, in these systems all the fingers are controlled collectively. The proposed method has the advantage of enabling the fingers to be controlled independently on the basis of the strength of the muscular activity of each finger. This is achieved by incorporating the results of the finger operation identification into the target angle of the prosthetic finger. Both of the proposed methods for the myoelectric prosthetic finger control have been evaluated experimentally.

目次

第１章 緒論

本章では，研究の背景と目的について述べる．

1-1. 研究背景

筋電義手は，腕の機能再現を目的とした電動義手の１つである．筋肉の収縮時に発生する筋電位信号を測定して制御を行っている．筋電位信号は皮膚表面からでも微弱に測定することもできるため，その信号から利用者の動作意図を識別することによって動作を義手が再現する．近年では，筋電計からの特徴量抽出方法や動作識別の方法も多く研究がなされており，実際の現場の利用者だけでなく市販化に向けた研究開発にも注目が集まっている．2014 年には FDA（アメリカ食品医薬品局）によって筋電義手が承認された，今後さらに市販への道が大きく開かれると考えられる．

近年の筋電義手に関する研究は，次のように大別できる．

① 筋電計から得られる表面筋電位信号をどのような処理を用いて識別や制御に取り入れるかに関する，信号処理に関する研究.

② 数多くある識別方法の中からどのような識別器をどのように筋電義手システムに取り入れるかを考える研究，識別器に独自に工夫を加えて機能を向上させる研究.

③ 識別された動作を義手に再現する制御方法や義手自体の設計に関するハードを考慮した研究.

研究では，筋電計には侵襲性のない表面筋電位(Surface electromyography: SEMG)が広く用いられている．動作の識別にはニューラルネットワーク，自己組織化マップなどが多く見られている．

1-2. 問題定義

今日の義手においては 2 種類の制御方式がよく見られている．それを説明するために図 1 に代表的な技手を示す.

Figure 1 SAWAMURA Prosthetics and Orthotics Service Co., LTD.
Touch Bionics IPO 「i-LIMB Pulse」

まず 1 つは，義手の力加減による制御を行ったものです．図 1 の左で示した義手は腕の表面につけた 2ch の電極からそれぞれ力加減を測定し一方が強ければ義手が閉じ，他方が強ければ義手が閉じるといった仕組みを取り入れている．機構もそれに応じてシンプルに設計されており，自由度は開閉に用いる 1 自由度のみとなっている．しかしながらヒトの腕のような多くの自由度を再現することはできない．

もう 1 つは，装着者の動作意図を多数電極から読み取りそれに応じた動きを再現するものです．図 1 の右で示した義手は複数の自由度を持っており指先で摘んだり，物を握りこんだりすることが可能となっている．こちらの制御方法では動作の識別のみによる比例的な指角度制御がなされることが多く，力加減を調節するといった微細な制御は少ない．

市販の義手の抱える問題点も考えると，ヒトの筋肉は栄養状態や疲労によって影響を受けるように日によって変化する．時にはそれにあわせて義手のシステムに調整させることも必要になってくる．多くの義手企業は義手を売ってそのままにせずサポートを充実させて対応を行っており，義手販売会社のない国で普及率を上げる大きな妨げとなっている．義手自体の価格も高く，電極数が多く複雑な制御が可能であるものほど高いように感じられる．

1-3. 義手研究

近年の筋電義手でなされている研究について少し触れる．

田村[1]らは筋の周波数特性を数値化するために必要なフーリエ変換を行わない動作識別方法を提案し，筋電位信号を周波数解析しなくても識別率を高い水準にすることができる信号処理方法の 1 つを示した．斎藤[2]らは指動作識別において多く使われている階層型ニューラルネットワークと自己組織化マップとの識別性能の比較を行っている．古川[3]らは提案した手の動作識別法を前腕切断者に実際に適用して実用性を実証している．原田[4]らは手首から先の切断者を対象とした拇指と示指の 2 本の指を搭載したロボットハンドを

製作し，拇指と示指の屈曲・伸展の 4 動作の識別により，ロボットハンドを筋電義手として駆動することに成功している．

識別信号となる表面筋電位信号の大きさは，個人や電極の装着部位によって変動し，識別に大きく影響する．棒谷[5]らは不特定多数の使用者に対応できる識別に関する研究を行っているが，本研究では使用者毎に識別器の学習を行う．一般的には使用者が異なる場合には使用者毎に学習用のデータを取得して識別器の学習を行う必要がある．

末光[6]は，学習に必要なデータ作成と処理の時間を考慮して，識別を行うための特徴量として，筋の活動状態を反映するといわれる積分筋電位(Integrated Electromyography: IEMG)信号を採用しており，本研究でも IEMG を説く跳梁として扱う．井部[7]らは，動作開始直後の立ち上がり時を避け，識別を行うタイミングに動作後に出てくる IEMG がピーク値を迎えるときとしている．これにより，同じ動作に対する特徴量のばらつきが小さくなり，識別精度が高くなると考えられる．

識別器には識別能力がすぐれているといわれるサポートベクターマシン(Support Vector Machine: SVM)を用いる．基本的なSVMでは，簡潔なニューマルネットワーク(Neural Network: N.N.)[8]や自己組織化マップ(Self-organizing Map: SOM)で見られる3種類以上の識別を行うことができない．そこで，3つのSVMを組み合わせた独自の識別器を構築することによって，拇指と拇指以外の4指の屈曲・伸展動作に加えて，全指での屈曲・伸展動作の計6動作を識別する．

先行研究の多くは識別された動作に対して予め用意しておいた義手の動作を単純に実現しているに過ぎない．

1-4. 研究目標

前項まで出述べた義手の研究背景に問題点や先行研究を考えて本研究では課題を解決するために以下のことを研究目標とする.

1) 義手には費用がかからないように少ない電極で多くの動作を実現する.
2) 筋力の持続時間および筋力の度合いを義手の制御に取り入れる.
3) 状況が日々変わっていく筋肉に対応できるように筋電の測定箇所，動作数と種類，多自由度の義手に対応するような制御を考える

電動義手の制御には，筋力を維持する時間とその活動量を表す IEMG を取り入れた 2 種類の制御方法を提案する．1 つは指動作における筋力の持続時間を考慮したものである．しかし，これだけでは義手の制御に力加減が考慮されておらず，もう 1 つは持続時間に加え，筋力の度合いを考慮したものとした．さらに，後者においては動作に対応した筋のみの IEMG を取り入れた手法と，一本の指を想定して屈筋と伸筋の IEMG の差分を取り入れた手法の 2 つを考える.

構築したSVMによる識別器の性能を検証する実験を行うとともに，IEMGを制御に取り入れた義手の指角度制御実験を3つの制御方法でそれぞれ行った.

1-5. 本稿の構成

以上説明した研究背景，問題定義，義手研究，研究目標から本稿の章立てを以下のようにして構成した.

第 2 章では本研究に使用した機器の説明を記載する．第 3 章では対象とした指動作に測定部位，抽出した特長量と識別に用いた SVM の説明をする．第 4 章では提案する指動作識別について説明を記載した後に，第 5 章にて動作識別による制御と力み度合いを取り入れた制御を行う．そして第 6 章で結論を述べる.

第２章 実験環境

本研究で使用する機器及び装置の説明をする.

2-1. 筋電計

筋電位を測定する方法には 2 種類あり，1 つは皮膚表面からでも測定する表面筋電位(SEMG)ともう 1 つは筋肉の電位を直接測定する針筋電というものがある. 針筋電計は電極が同心型針電極となり，針の先端役 0.5mm の中にある筋繊維から発生する活動電位を捉えることが可能である. そのため電極としては非常に小さく体内にある筋をピンポイントで測定することができ，高い空間分解能で識別できる. しかしながら針を体内に刺入するため人体に対して侵襲性がある. 本研究では被験者が複数いることを考えて，痛みを伴わない表面筋電計にて測定を行う.

表面筋電位は皮膚表面からでも測定できるため，侵襲性が無い. 多くの人や測定部位に適応することができる. しかし，貼りつけた皮膚の位置によっては目的の筋の活動量だけではなく，目的とした筋以外の周囲の筋活動量も含まれてしまう. さらに生体信号であるため，被験者毎に筋電位特徴が異なることや，同一の被験者においても測定部位の姿勢に応じて時々刻々と差異が生じる. よって使用の際には十分に考慮しなければならない.

本研究では図 2 に示す Biometrics 社製のアンプ付筋電位測定電極 SX230W EMG Amplifier を用いており，電極から DHK 社製中継ボックス，アイソレータ PH-2501/8 を介し電圧を測定する.

Figure 2 SEMG measurement device and electrode

2-2. 測定機器

表面筋電計のアンプからの信号を AD 変換することによって PC に取り込む，AD 変換器には Inteco 社製の RT-DAC4/PCI を使用する．様々な機能を持つコンピュータボードであり，RT-CON Toolbox を用いて MATLAB/Simulink のプログラムへとデータを送ることができる．PC の制御周期は 1msec とする．図 3 にはボードの外装と図 4 にはボードのコネクタの分布を示す．

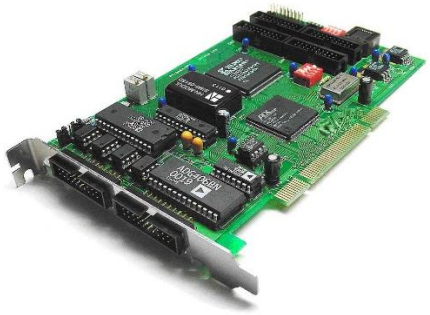


Figure 3 RT-DAC4/PCI MultiI/O Board

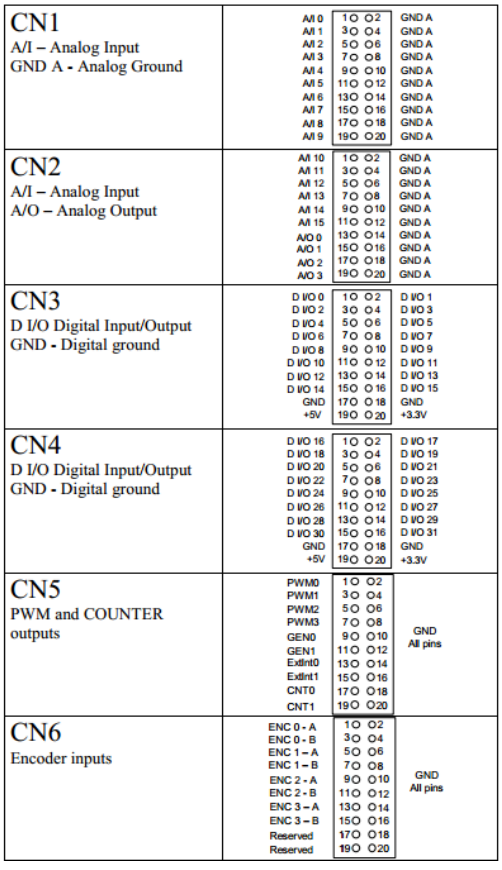


Figure 4 RT-DAC4/PCI I/O connectors

2-3. Matlab/simulink

MATLAB は工学分野や科学分野などの様々なデータ解析，シミュレーションなどにおける行列計算，ベクトル演算，グラフ化などを行う Math Works 社が提供するソフトウェアである． MATLAB はインプリンタ形式のプログラミング言語である．インプリンタ型はコンパイラ型に比べて，プログラムを単独で実行可能であるため，容易にプログラムの変更・修正を行うことが可能である．

MATLAB がプログラミング言語であるのに対し，Simulink はブロック線図を用いたシステムを構築するソフトウェアである．Simulink は MATLAB のプロダクトファミリの 1 つであり，MATLAB との互換性がある．また Simulink におけるインターフェイスはブロックダイアグラムツールとブロックライブラリから成り，視覚的にシステムを構築できるため，熟練したプログラム言語を必要としない．論理演算や算術演算，伝達関数のブロックを組み合わせることで数学モデルのシミュレーションや信号処理など様々な環境構築ができる．また，既存のブロックだけでなく，MATLAB で作成したプログラムや C 言語，Fortrun, BASIC 言語などで作成したプログラムをブロックとして Simulink のブロック線図に加えることができる．MATLAB/Simulink には Toolbox というオプションを加えることで信号処理を始めとした様々な機能の拡張を行うことができる．

第３章 対象動作および測定筋の選定

本研究で対象とした動作と，特徴量の測定方法および識別器に用いた SVM についての説明を行う．

3-1. 指動作

本研究では，前腕部に動作筋が集中する指動作に着目し，4 か所の部位から測定した表面筋電位(Surface Electromyography: SEMG)を利用することにより，拇指の屈曲・伸展，拇指以外の 4 指の屈曲・伸展，全指での屈曲・伸展の 6 動作を識別する．識別動作の様子を図 4 に示す．これらは健常者が容易にできる動作である．全 6 動作は二次元のベクトルとして表し，それぞれの成分は 1 行目が拇指の状態を示し，2 行目が 4 指の状態を示す．ここでは，1 は屈曲，-1 は伸展を意味する．

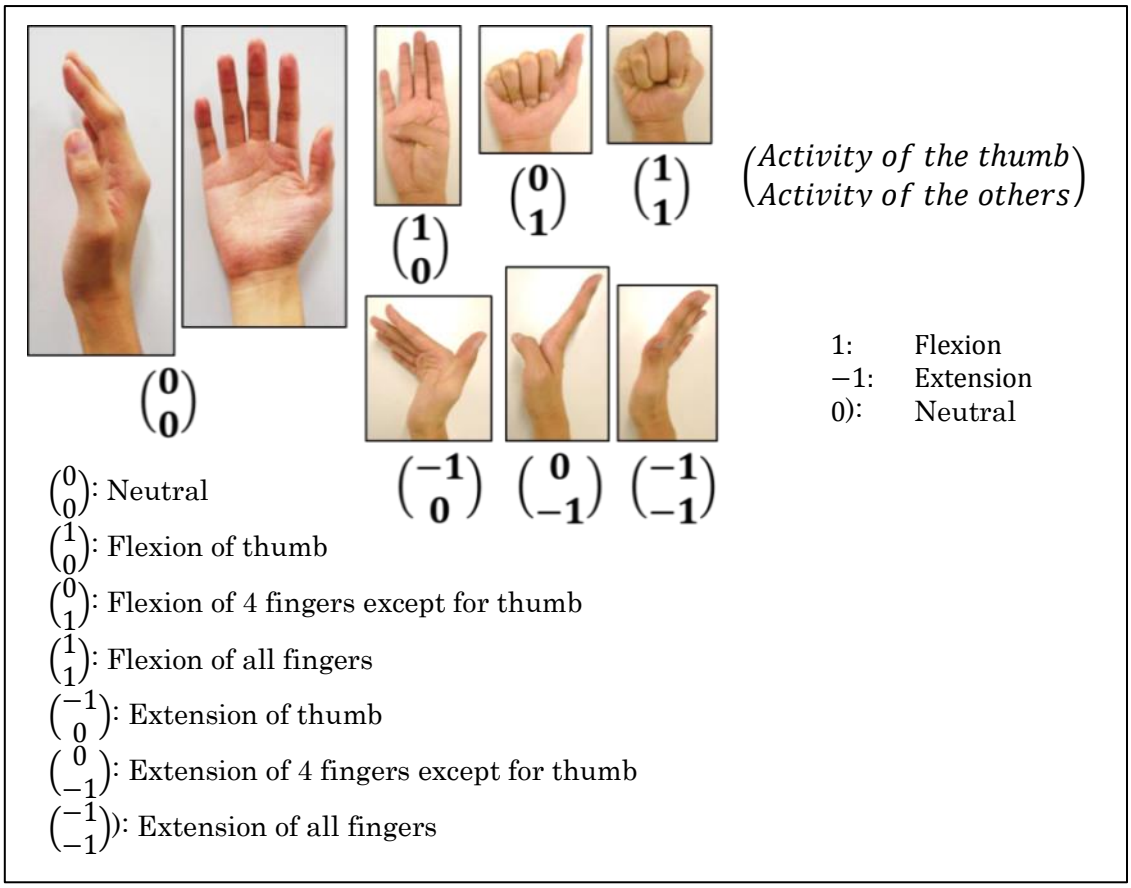
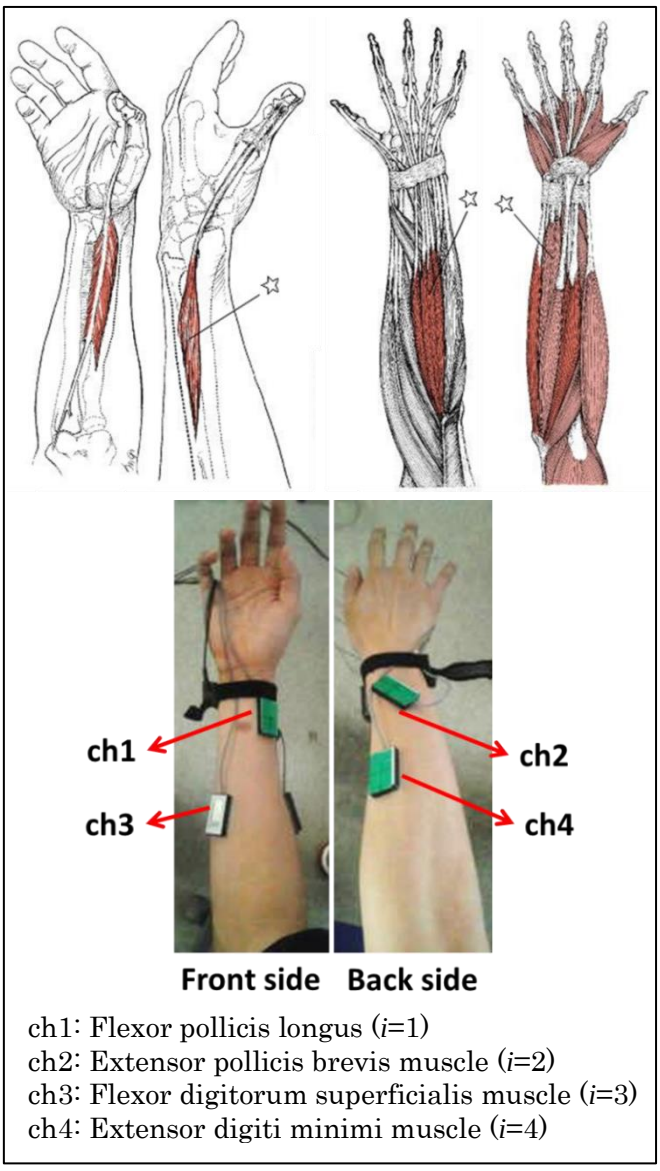


Figure 5 Targeted finger motions

3-2. 表面筋電位の測定

表面筋電位はそれぞれの指動作を行う動作筋から測定する．図２に本研究で使用した 4 つの動作筋（ch1: 長拇指屈筋，ch2:短拇指伸筋，ch3:浅指屈筋，ch4:小指伸筋）における SEMG の測定位置を示す．これらはそれぞれ前項で示した動作を行うときに利用する筋肉のうちの 1 部となっている．筋電計貼り付け時にはほかの筋肉の信号を極力ノイズとして入らずに，筋の収縮時には SEMG に大きく反応が見られる必要がある．個々によって場所も大きさも変わるため注意が必要である．



ch1: Flexor pollicis longus (i=1)
ch2: Extensor pollicis brevis muscle (i=2)
ch3: Flexor digitorum superficialis muscle (i=3)
ch4: Extensor digiti minimi muscle (i=4)

Figure 6 Positions of electrode

3-3. 積分筋電位

積分筋電位(IEMG)は，皮膚表面に貼り付けた筋電計により測定される SEMG の絶対値を一定区間ごとに積分することによって得られる値であり，筋の活動状態全体を反映する指数と考えられている．図 7 には例として弱い力みと強い力みをした時の ch3 の IEMG と SEMG の様子を示す．IEMG と SEMG の関係は筋が活動すれば，つまり，SMEG に大きな振幅が見られればそれに応じた値を示す．筋の収縮が強いほど大きい値，弱ければ小さい値となる．(1)式に IEMG の算出式を示す．IEMG にはさまざまな定義があり数値や単位系が変わってくるが，本研究では以下のように扱う．

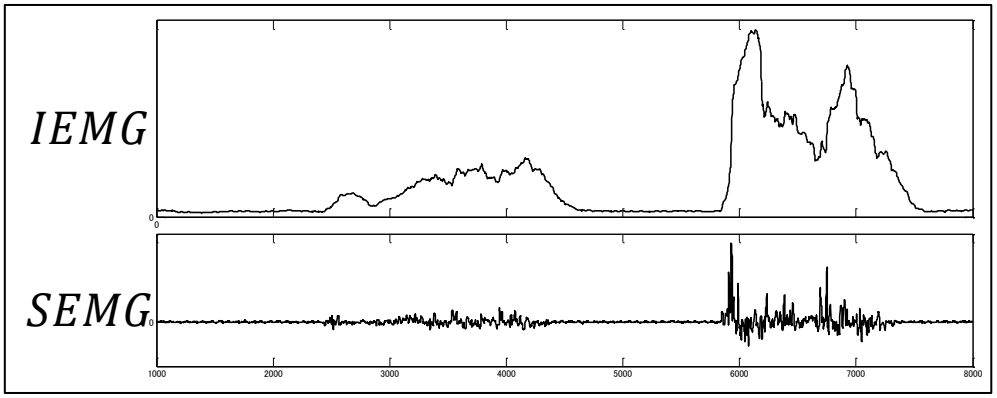


Figure 7 signal of IEMG and SEMG

$$IEMG_i = \int_{t-\Delta t}^{t} |SEMG_i|\, dt \quad (i = 1\sim4) \tag{1}$$

ここで，$SEMG_i$は各測定部位の表面筋電位の値であり，本研究では 1kHz 毎に測定する．$i(=1\sim4)$は測定部位に対応するチャンネルを表す．Δtは積分区間を表し，実質的にはサンプル数を意味する．本研究では積分区間を0.256sとし，サンプル数を 256 とした．

本研究では計算量をすくなくするために．下式のように計算をした．

$$IEMG_{i_{(t)}} = \int_0^t |SEMG_i|\, dt - \int_0^{t-\Delta t} |SEMG_i|\, dt = \int_0^t \left(\left|SEMG_{i_{(t)}}\right| - \left|SEMG_{i_{(t-\Delta t)}}\right|\right) dt$$

$$= IEMG_{i_{(t-0.001)}} + \int_0^{0.001} \left(\left|SEMG_{i_{(t)}}\right| - \left|SEMG_{i_{(t-0.001)}}\right|\right) dt \quad (i = 1\sim4)$$

ここで，0.001 はサンプル時間となっている．特に離散的に IEMG を扱う場合において

計算量の大きな簡略化がなされている．本研究で実際に使用した IEMG の算出システムを図 8 に示す．10 分間の測定を行ったが，値の発散は見られなかった．SEMG は全派正流化する必要があるため測定 1 秒間は SEMG の平均値を取りその値を引くようにしている．

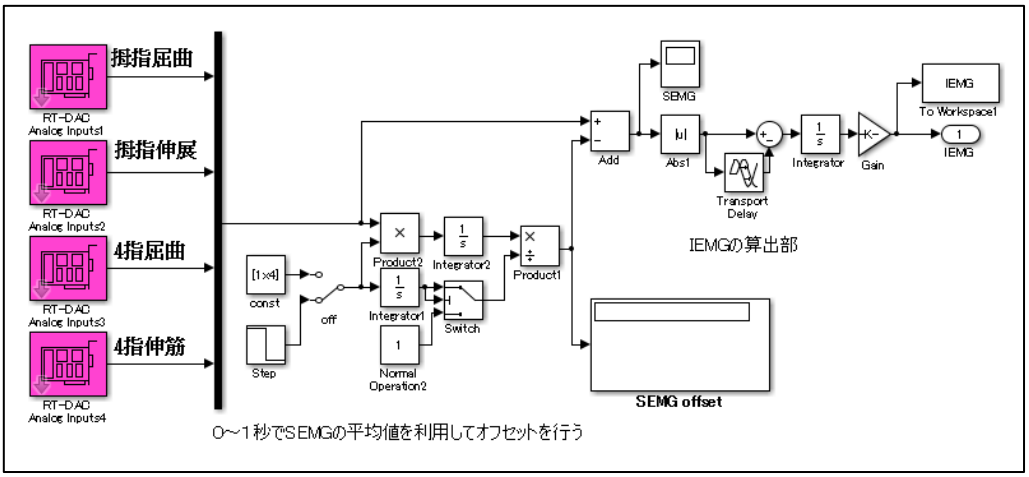


Figure 8 IEMG_out subsystem

3-4. サポートベクターマシン

サポートベクターマシン(SVM)は，教師あり学習により識別を行う手法の一つで，教師信号から各データとの距離が最大となるマージン最大化によって識別の境界となる超平面を求め，領域を分割する．線形分離不可能な問題（二次元では直線，三次元では平面で分けられない問題）に対しては，入力値を高次元の特徴空間と呼ばれる内積空間に写像する方法がある．2 次元の線形分離不可能問題を 3 次元で線形分離するような例を図 9 に示す．実際には，写像関数を直接計算する代わりに，特徴空間の変数の内積（カーネル行列）を用いて最適な識別関数を構成できる．これはカーネル法(Kernel Trick)と呼ばれる．カーネル法によって，SVM は識別手法の中で最も識別性能が優れた方法の一つといわれるようになった．

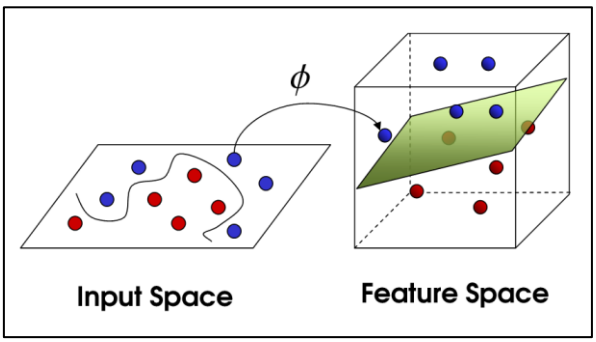


Figure 9 Conceptual diagram of pattern identification by SVM [9]

第 4 章 識別実験

本章では識別に用いた手法と識別の性能を検証した実験について説明する.

4-1. 特徴量ベクトル

安定した IEMG の値を特徴量にしたいため，動作開始直後の立ち上がり時を避け，立ち上がり後に IEMG がピークを迎えたときの値を測定する．このため，以下に示す手順で SVM の学習データと識別に用いる特徴ベクトルを作成する.

1) SEMG の正流化と IEMG の最低値を 0 にするために 2 秒間動作をせずに待つ.
2) 6 つの動作を各数回ずつ行い，4 つのチャンネル(ch)それぞれのIEMG の最大値を求める．この値を$IEMG_{max}$とする.
3) 測定を繰り返すうちに，動作開始を判断する目安として$IEMG_{max}$の 50%の値が適当と判断されたため, 拇指, 4 指, 全指の順に屈曲･伸展を行い, 関係する筋のch の IEMG 値が $IEMG_{max}$ の 50%を超え，尚且そのch がはじめてピークを迎えたときの全ch のIEMG の値を 4 次元の特徴ベクトルとして保存する. チャタリング防止の為動作間には IEMG が 20%未満になるまで待つ.

特徴ベクトルは以下のようになる．添え字は第 2 章で述べた ch を意味する．図 10 には特徴ベクトル保存に用いたプログラムを示す．背景が水色のブロック内（ファンクションプログラム）は付録 A に記載する．SVM_out は図 8 に示したものとなっている.

$$\begin{pmatrix} IEMG_1 \\ IEMG_2 \\ IEMG_3 \\ IEMG_4 \end{pmatrix}$$

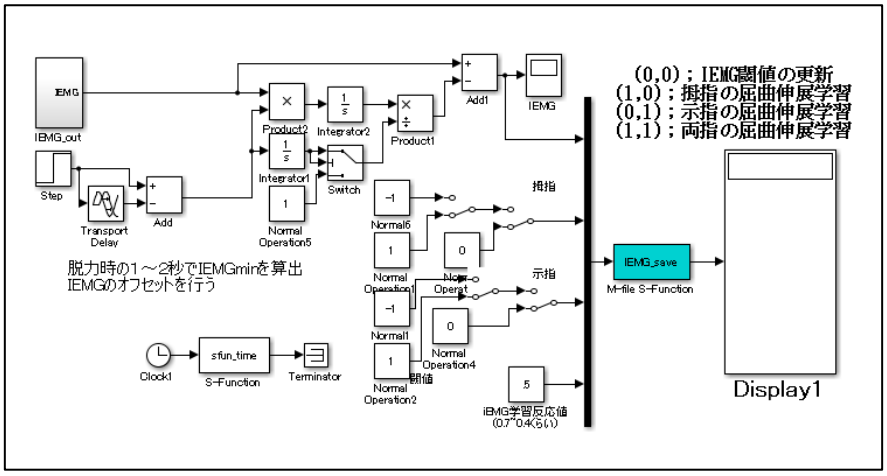
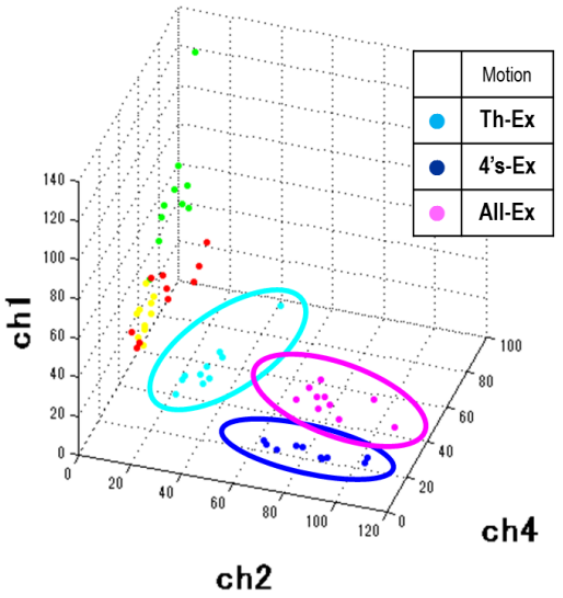
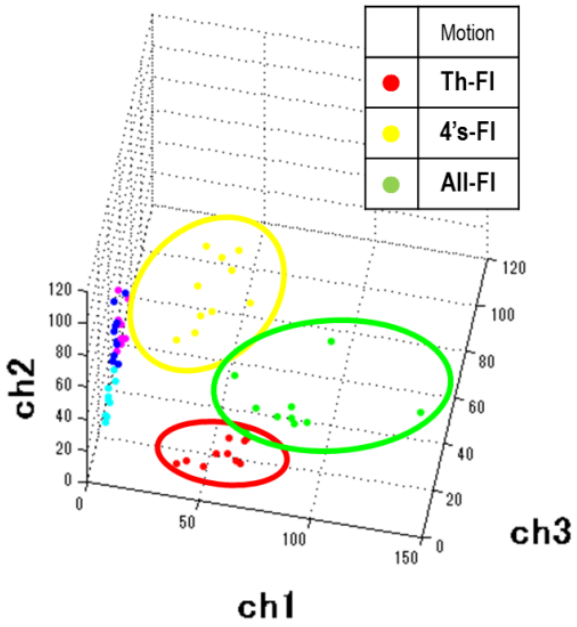


Figure 10 MATLAB/Simulink Program to save feature vectors

SVM を学習させるために，拇指，4 指，全指の屈曲・伸展それぞれ 10 回分，計 60 個の学習用特徴ベクトルを学習データとして作成した．図 11 に作成した特徴ベクトルの分布図を示す．可視化するために特徴ベクトルから 3 次元分を取り出したものを示す．全体的に見て，屈曲と伸展の区別が可能な分布になっているといえる．これより，提案した特徴ベクトルを用いて動作識別できることが期待できる．



(a) Extension (ch1, ch2, ch4)



(b) Flexion (ch1, ch2, ch3)

Figure 11 Distribution of feature vectors

4-2. 識別器の構造

第 3 章で述べたように, 一般的な SVM は入力信号に基づき 2 種類の領域に出力を分割する．つまり，6 つの動作を単純な SVM で識別することができない．そこで，本研究では 3 つの SVM を使用して 6 つの動作の識別を行う識別器を構築する．3 つの SVM ではそれぞれ以下の役割を持って動作を識別する.

・SVM-a：屈曲動作(1)か伸展動作(-1)かを識別
・SVM-b：拇指動作(1)か 4 指動作(-1)かを識別
・SVM-c：全指動作(1)かそれ以外(-1)かを識別

3 つの SVM の識別結果から対象者の行った動作が 6 種類のどの動作に分類されるかを割り出す．表 1 に示す指の状態を表す 2 次元のベクトルを構成する．これより，6 種類の指動作が識別される．提案した 3 つの SVM を用いた識別器による識別の流れを図 12 に示す.

指の状態を表す 2 次元のベクトルは，1 行目が拇指の状態 SVM_{out_1}，2 行目が 4 指の状態 SVM_{out_2}を表しており, 2 行合わせると全指の状態を表す. ベクトルの成分が 1 のとき屈曲，-1 のとき伸展，0 のときは脱力・無動作を表している.

Table 1 Identification by SVM based identifier

		SVM-a = 1	SVM-a = -1
SVM-c = 1		$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}-1\\-1\end{pmatrix}$
SVM-c = -1	SVM-b = 1 (Thumb)	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}-1\\0\end{pmatrix}$
	SVM-b = -1 (4 fingers)	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}0\\-1\end{pmatrix}$
Else (Neutral)		$\begin{pmatrix}0\\0\end{pmatrix}$	

$\begin{pmatrix}0\\0\end{pmatrix}$: Neutral
$\begin{pmatrix}1\\0\end{pmatrix}$: Flexion of thumb
$\begin{pmatrix}0\\1\end{pmatrix}$: Flexion of 4 fingers except for thumb
$\begin{pmatrix}1\\1\end{pmatrix}$: Flexion of all fingers
$\begin{pmatrix}-1\\0\end{pmatrix}$: Extension of thumb
$\begin{pmatrix}0\\-1\end{pmatrix}$: Extension of 4 fingers except for thumb
$\begin{pmatrix}-1\\-1\end{pmatrix}$): Extension of all fingers

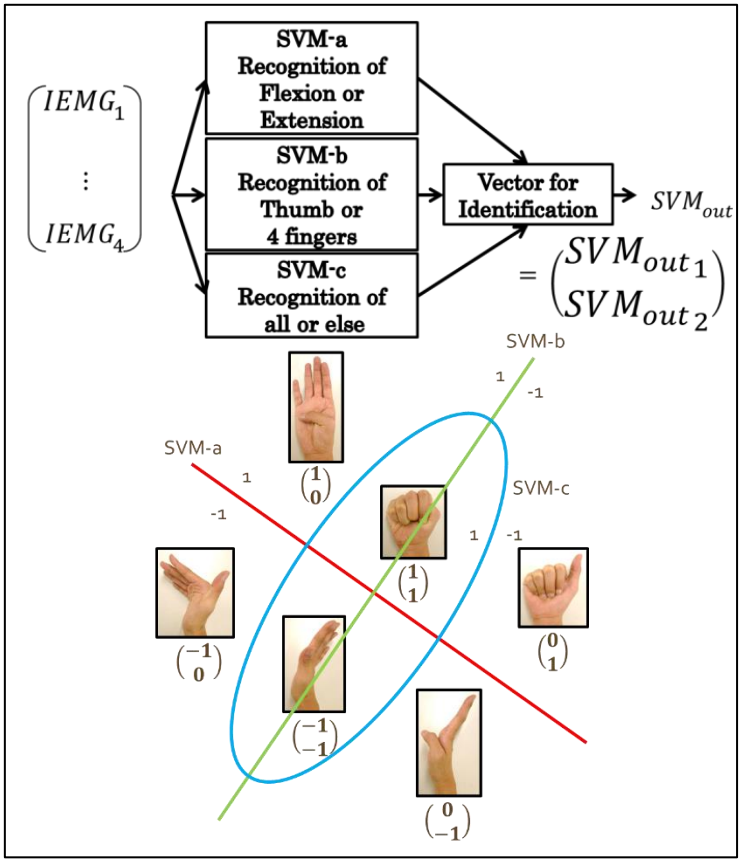


Figure 12 Structure of SVM based identifier

作成した学習データを用いてそれぞれの SVM を学習させる．学習のために必要となる教師信号は，SVM-a では屈曲を(1)，伸展を(-1)とし，SVM-b では拇指動作を(-1)，4 指動作を(-1)とし，SVM-c では全指動作を(1)，それ以外を(-1)とした．また，SVM-b では全指での特徴ベクトルは不要となるため，学習には学習データから除外して学習を行う．SVM については Information: Signals, Images, Systems: ISIS[10]にて公開されているツールボックスを利用して学習を行った．この際，カーネル行列には spline を用いた．

4-3. 識別実験

提案した識別器の検証の為に，識別実験を行う．被験者は 20 代の健常者男性 5 名である．それぞれの被験者の学習データを用いて学習させた SVM を用いた識別器により，オンラインで動作識別の検証を行う．本章で説明する手順で 4 次元の特徴ベクトルを作成し，識別させる．各動作 30 回，計 180 回分の動作を識別した．

被験者 5 名のうち 1 名は十分に動作の練習をしておりそれ以外の 4 名は筋電計の装着後およそ 30 分程度の時間が経過してから識別実験を行った．

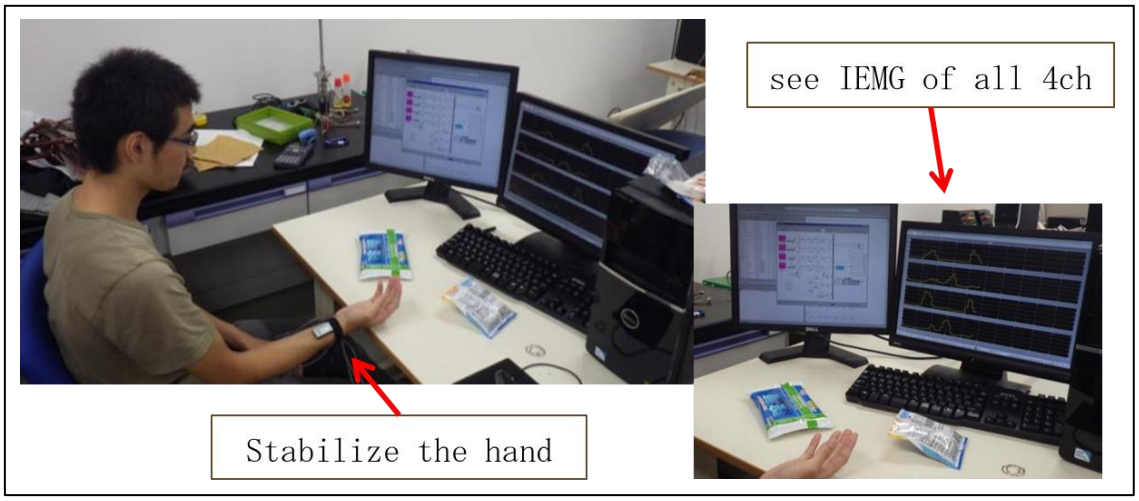


Figure 13 Saving feature vectors

4-4. 識別結果

各動作に対するオンラインでの識別率を表 2 に示す.

Table 2 Accuracy of identification [%]

Subject	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}-1\\0\end{pmatrix}$	$\begin{pmatrix}0\\-1\end{pmatrix}$	$\begin{pmatrix}-1\\-1\end{pmatrix}$	**Ave.**
A	100	90	93	100	100	93	96
B	87	80	70	100	90	63	82
C	100	97	97	100	57	97	89
D	93	77	83	100	90	57	83
E	80	67	77	83	100	100	84

全 5 名の被験者において識別率の平均値が 80%を越えることができた．被験者 A は充分に動作を繰り返し，動作に慣れているため，他の被験者に比べて高い値を示している．B～E の被験者に関しては，A に比べてそれぞれの動作に慣れていなかったため，識別率が低くなったと考えられる．対象動作を練習することによって識別率の向上が期待できる．
被験者ごとに指動作に対しても筋電位に個人差が見られるため，本研究の識別器はここに作成することが望ましいと考える．

第５章 筋電義手制御

識別結果に基づき，オンラインにてロボットハンドの指角度制御を行う．本研究では以下の識別結果を利用した制御（Method 1）と筋肉の力み度合いを考慮した制御（Method 2）の 2 種類の制御方式を提案する．

1) 指に掛けている筋力の持続時間に基づいて義手の指の角度を変化させる．（以降 Method 1 と記す）

この方法では掛けている筋力の強弱が反映されず，指の目標角度は筋力の度合いによらず同じ値になる．そこで 2 つ目の制御方式として，筋力の度合いも考慮したものを考える．

2) 指に掛けている筋力の度合いと筋力の持続時間に基づいて，義手の指の角度を変化させる．（以降 Method 2 と記す）

さらに Method 2 においては，動作に対応した筋の IEMG を取り入れた手法（Method2-a）と，屈筋と伸筋の IEMG の差分を取り入れた手法（Method2-b）の 2 つの方法を考える．

5-1. 制御ロボットハンド

本実験では，ワイヤーを巻き取ることにより指の屈曲を行い，ゴムの弾性力により指の伸展を行うロボットハンド[4]を義手として使用する．図 14 にロボットハンドの外観を示す．なお，本ロボットハンドは拇指と示指の 2 本の指しか持たないため，拇指を拇指動作，示指を 4 指の動作にて代用する．

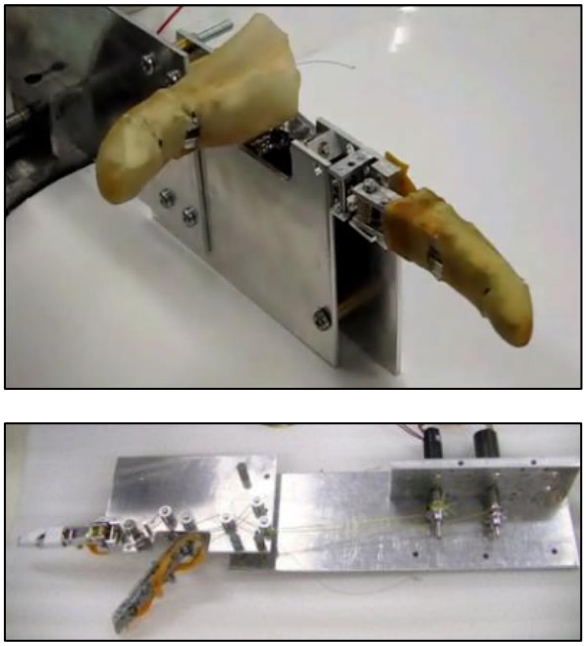


Figure 14 Robot hand

義手自体は指の全関節を屈曲できる機構になっているが，指の根元である MP 関節の目的角度の制御を行う．拇指の MP 関節の角度をθ_1，示指の MP 関節の角度をθ_2として，指を伸ばした状態を下限値$\theta_{min_j} = 0$ rad $(j = 1,2)$, 指を曲げきった時の値をθ_{max_j} $(j = 1,2)$とする．使用するロボットハンドでは$\theta_{max_j} = \frac{\pi}{2}$ rad $(j = 1,2)$となっている．ロボットハンド 3D モデルを用いて示指角度の座標を図 15 に示す.

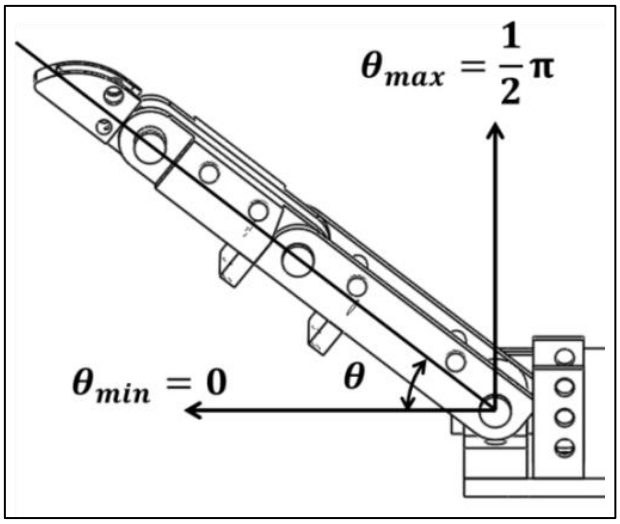


Figure 15 Angle of finger of robot hand (index finger)

5-2. 動作時間による制御 Method 1

操作者が動作後に対象となる筋の筋力を維持することによってIEMGの値を維持することができる．筋力を維持している時間に基づいて義手の指の目標角度を設定することにより，操作者は義手の指の角度を任意に制御することができる.

指を伸ばした状態から曲げきった状態まで動かす時に掛かる時間を$1/K_1$ [sec]としたとき，義手の指の目標角度θ_jを(2)式で与える.

$$\theta_{refj} = \theta_{max_j} \int K_1 \, SVM_{out_j} \, dt \quad (j = 1,2) \tag{2}$$

ここで，j=1 は拇指，j=2 は示指を示している．また，$0 \leq \theta_j \leq \frac{1}{2}\pi$とし，指の角度が上下限に達した場合は限界値を超えないように設定する．$SVM_{out_j}(j = 1,2)$は，指ごとの識別結果，K_1は指の動作速度を示す定数である．今回はロボットハンドの指動作が 1 秒で終わるように$K_1 = 1$とした.

また，全 4ch のIEMGが$IEMG_{max}$ の 20%に収まった時を脱力・無動作時とみなしてこれ

を動作終了とする．動作終了とみなされない限り別動作の識別は行わず，動作開始時の動作を継続しているものとする．

オンラインでの制御実験に用いた MATLAB/Simulink のブロック線図を示す．図 16 でプログラムの全体を示す．

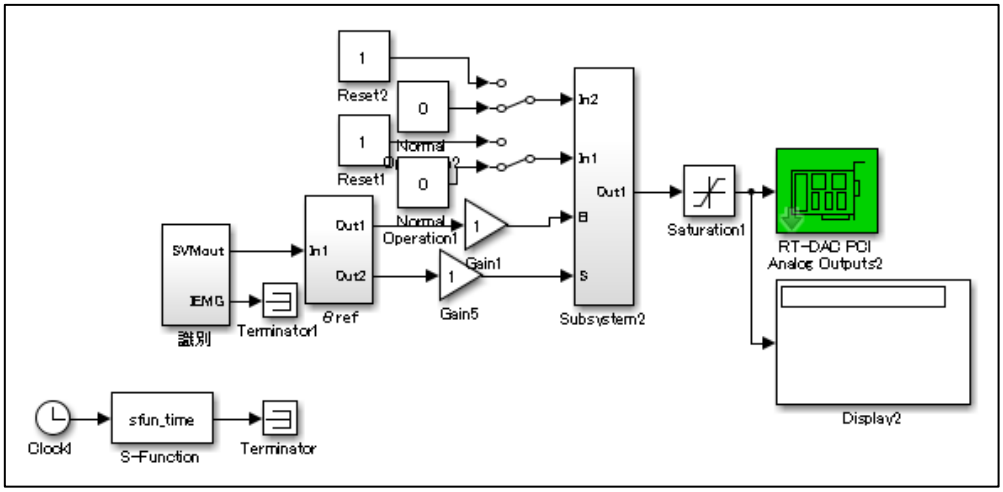


Figure 16 MATLAB/Simulink control using Method 1

ここでブロック内の Subsystem の識別を図 17 に，目的角度算出部分 θ_{ref} を図 18 に示す．

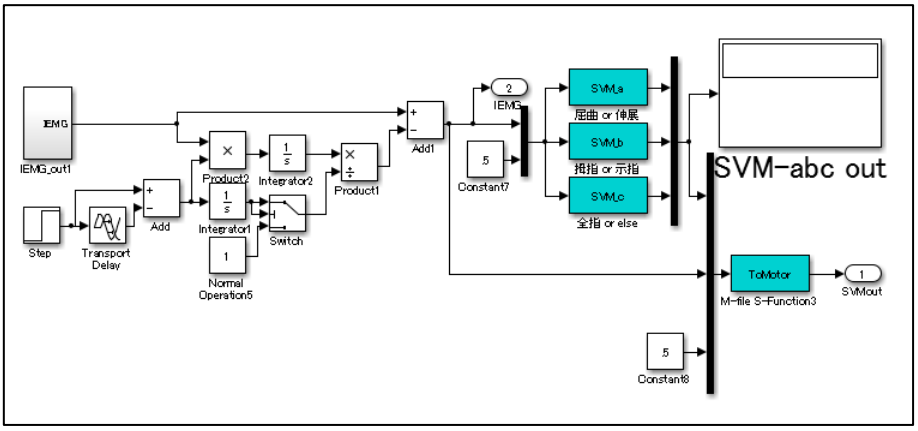


Figure 17　Identification subsystem

ここでの IEMG_out は図 8 のものを使用している．付録 B には 3 つの SVM を学習させてその結果を一時保存 CSV 形式で保存するプログラムを示す．添付 C には識別する IEMG 信号を識別する SVM-a，b，c での識別と，添付 D には識別結果を 2 次元ベクトルに統括するファンクションの内部を記載する．

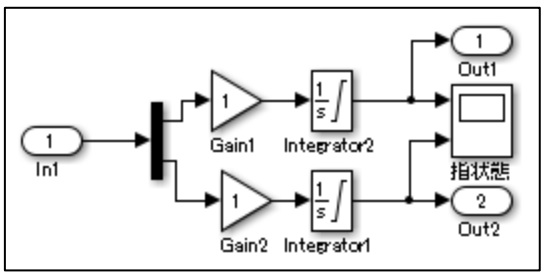


Figure 18　θ ref subsystem of Method 1

Method 1 を検証するために，ロボットハンドの制御実験をそれぞれの方法にて行った．24 歳の健康な男性 1 名を被験者として，算出された指の目標角度にロボットハンドの指の角度が追従するように，ロボットハンドのモーターを PID 制御器により制御した．10 秒間で拇指屈曲⇒拇指伸展⇒4 指屈曲⇒4 指伸展を行ったものを図 19 に示す. 上のグラフが 4ch 分の IEMG であり，下のグラフが指ごとの目標角度θ_{refj}と実際にロボットハンドの指を駆動しているモーターから推定した指の角度となっている．

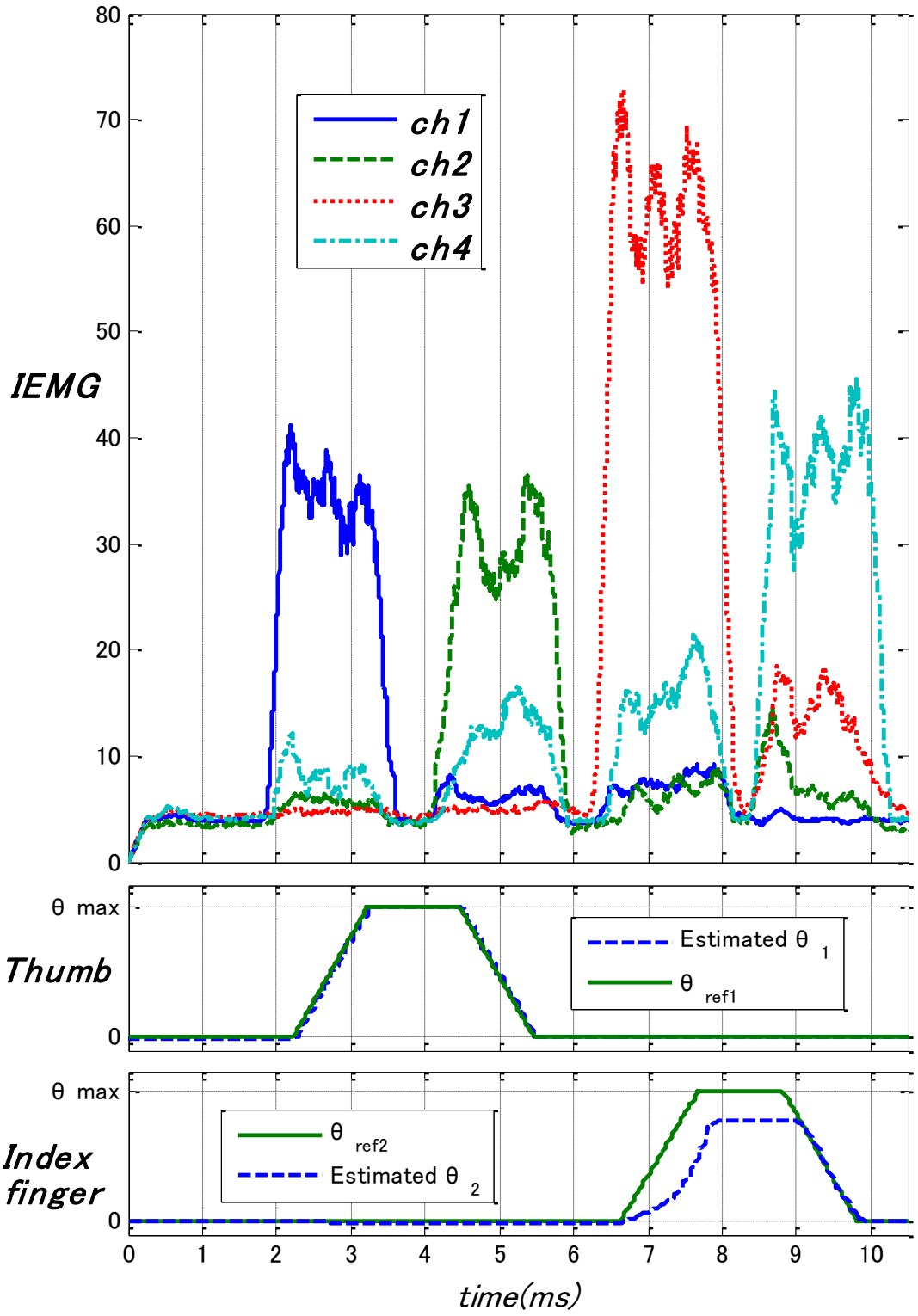


Figure 19 Results of finger control using control Method 1

5-3. 力み度合による制御 Method 2-a

Method 1 では，指に掛けている筋力の強弱によらず指の目標角度が決まる．筋力の度合いを指の目標角度に反映させるため, 前節の方法に筋の活動量を表すIEMG の値を取り入れて義手の指の目標角度を定める．Method 2-a では動作を行うときに使用する筋の活動量を操作者の意図として制御に取り入れる．

測定対象の筋肉ごとにIEMG の最大値が異なるため，IEMGの値が 0~1 となるように，次式により ch ごとにIEMG の正規化を行う．

$$NIEMG_i = \frac{IEMG_{i_{now}} - IEMG_{i_{min}}}{IEMG_{i_{max}} - IEMG_{i_{min}}} \quad (i = 1\text{–}4) \tag{3}$$

ここで，$IEMG_{i_{min}}$と$IEMG_{i_{max}}$ は学習データ作成時に得られるIEMG の最小値と最大値，$IEMG_{i_{now}}$は現在の IEMG の値，$NIEMG_i$は正規化された IEMG の値である．

SVM 識別器の結果から，(4)式によりそれぞれの動作に対応した IEMG を抽出し，$IIEMG_j$ $(j = 1,2)$とする．

$$IIEMG_j = \begin{cases} NIEMG_{2j-1} & \text{if } SVM_{out_j} = 1 \\ NIEMG_{2j} & \text{if } SVM_{out_j} = -1 \\ 0 & \text{if } SVM_{out_j} = 0 \end{cases} (j = 1,2) \tag{4}$$

ここで，$IIEMG_j$は動作に対応した ch が選ばれる．j=1 は拇指，j=2 は示指を示している．$IIEMG_j$を取り入れて，指の目標角度を次式で与える．各指に対応した ch の$NIEMG_i$が選ばれるようになっている．

$$\theta_{refj} = \theta_{max_j} \int_{t-\Delta t}^{t} K_2 \cdot SVM_{out_j} \cdot IIEMG_j \, dt \; (j = 1,2) \tag{5}$$

ここで, j=1 は拇指, j=2 は示指を示している. K_2は指の動作速度の係数であり, 今回は$K_2 = 1$とした．K_2の値は$IEMG_{max}$を出すように力を出し続けて技手の指が 1 秒で曲がるような値なので，多少遅めに設定している．操作者がなれれば大きい値にすることが望ましい．使用したプログラムを図 20 に示す．Method 2-b ではθ_{ref}のサブプログラムのみ異なる．図 21 には Method 2-a でのθ_{ref}サブシステムを示す.

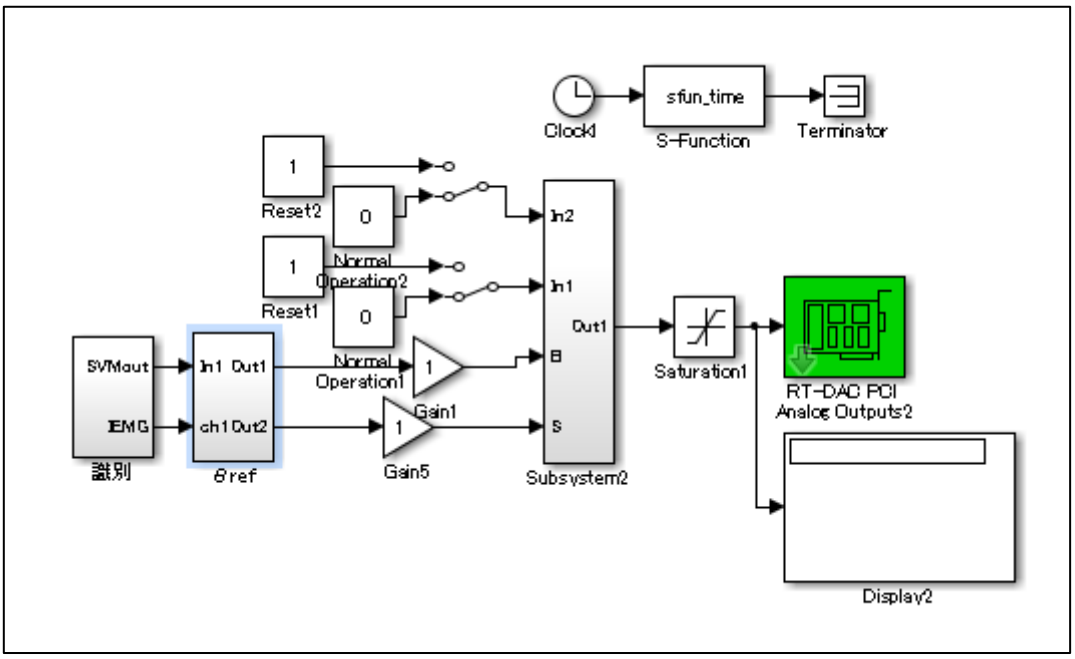


Figure 20 MATLAB/Simulink control using Method 2

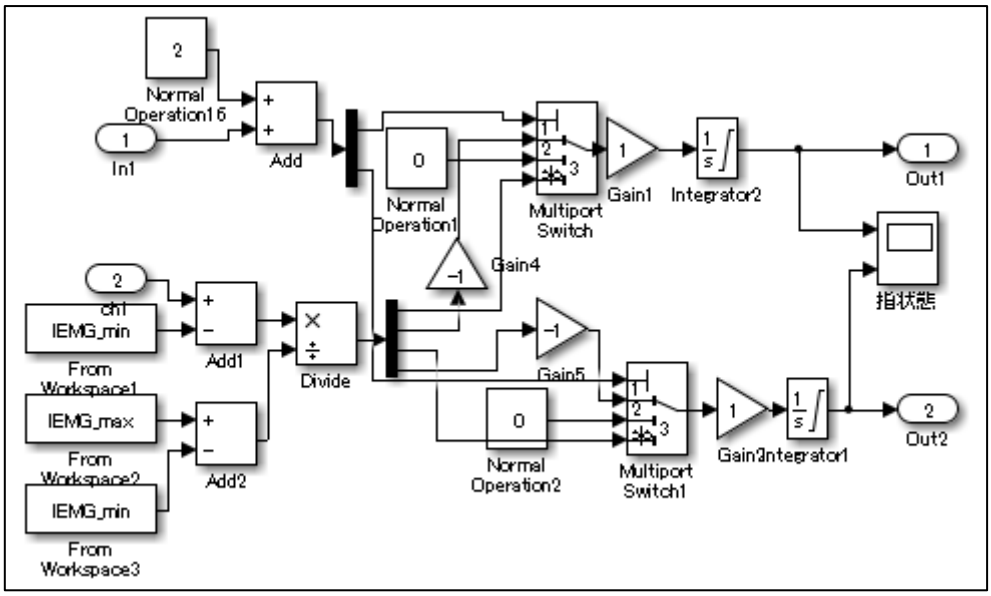


Figure 21 $\boldsymbol{\theta_{ref}}$ subsystem of Method 2-a

Method 2-a での制御を図 22 に示す．10 秒間で全指屈曲⇒全指伸展を 2 回繰り返し，1 回目の屈曲には拇指の力を強め，2 回目では 4 指の屈曲を強めた．

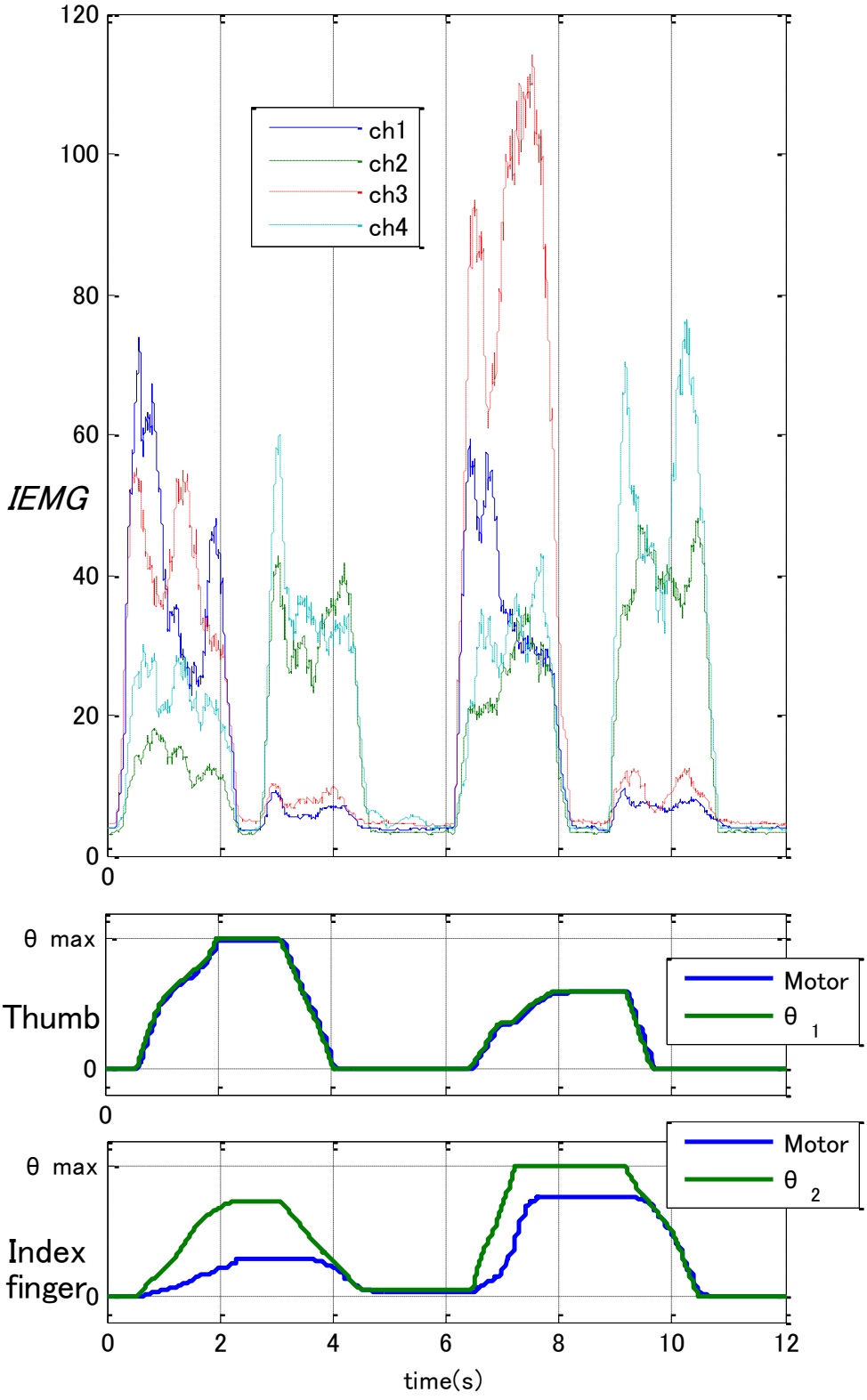


Figure 22 Results of finger control using control Method 2-a

5-4. 力み度合による制御 Method 2-b

1 つの指関節を動かすのに，屈筋と伸筋が使われる．そこで，屈筋と伸筋による IEMG の差分，すなわち差分積分筋電位を用いて制御を行う．差分積分筋電位$DIEMG_j$ $(j = 1,2)$を次式により算出する．

$$DIEMG_j = \left|NIEMG_{2j-1} - NIEMG_{2j}\right| \quad (j = 1,2) \tag{6}$$

$DIEMG_j$を取り入れて，指の目標角度を次式で与える．

$$\theta_{refj} = \theta_{max_j} \int_{t-\Delta t}^{t} K_3 \cdot SVM_{out_j} \cdot DIEMG_j \, dt (j = 1,2) \tag{7}$$

ここで, j=1 は拇指, j=2 は示指を示している．K_3は指の動作速度の係数であり，今回は$K_3 = 1$とした．ここで利用した MATLAB/Simulink を図 23 に示す．

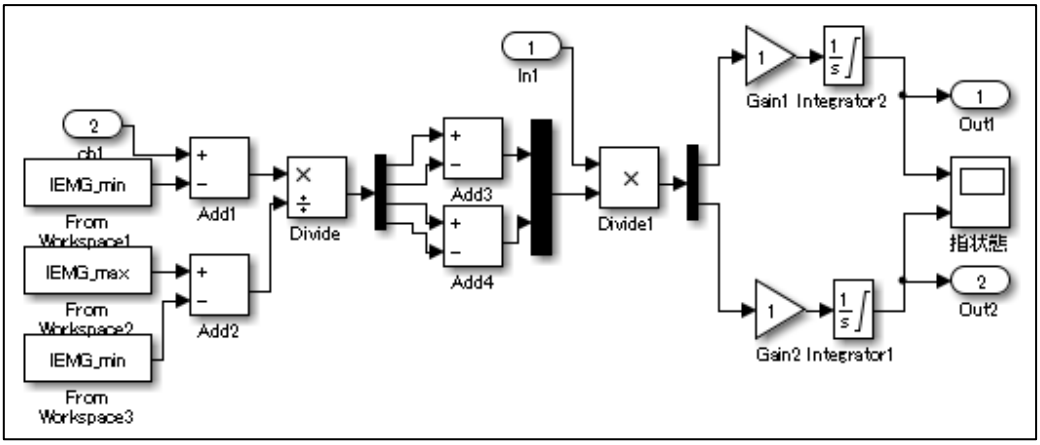


Figure 23 $\boldsymbol{\theta_{ref}}$ subsystem of Method 2-b

Method 2-a と同様に Method 2-b での制御を図 24 に示す．10 秒間で全指屈曲⇒全指伸展を 2 回繰り返し，1 回目の屈曲には拇指の力を強め，2 回目では 4 指の屈曲を強めた．

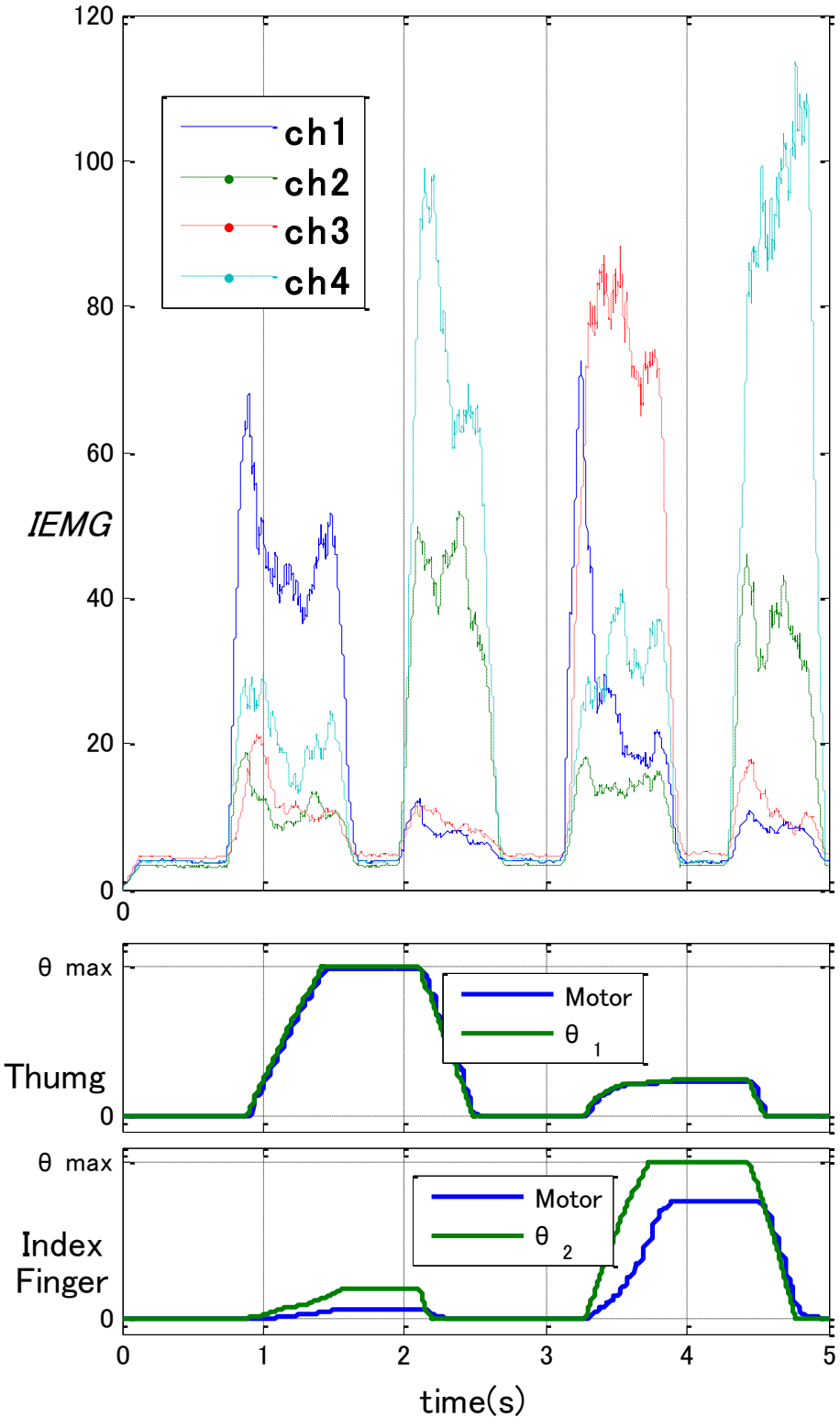


Figure 24 Results of finger control using control Method 2-b

5-5. 考察

Method 1 の制御結果である図 19 から，指の目標角度が筋力の持続時間に比例して変化していることがわかる．実験ではθ_{maxj}まで指を曲げる動作を行ったが，動作の途中に被験者が任意に脱力することによって,指の角度を途中で止めることもできる.Method 1 では，被験者がロボットハンドの両指を同時に曲げようとすれば，拇指と示指は IEMG の大きさに関係なく同じ速度で屈曲してθ_{maxj} に達するようになっている．

図 22 と図 24 では，両方とも IEMG の値によって拇指と示指の目標角度の変化量，および目標値が変わっていることがわかる．本実験では全指の屈曲・伸展動作を行ったが，拇指と示指の単独動作でも，力加減によってそれぞれの指の目標角度が変わり，動作速度も変わる．

Method 2-b の制御実験では，Method 2-a に比べて拇指と示指間での目標値の差分が大きくなっている．これより，Method 2-b の方が利用者の強弱の意図をより反映していると考えられるので，全指動作には Method 2-b の方が適していると言える．これは屈筋と伸筋の意図しない筋の力みが打ち消し合っているためと考えられる．指をそれぞれ動かすときには Method2-a 全指動作においては Method2-b が向いてると考える．

第６章 結論

本研究では，筋電義手制御のための指動作の識別方法として，3 つの SVM により構成される識別器を提案し，4ch 分の IEMG から 6 種類の指動作の識別を行った．識別実験において，4 人の被験者に対して筋電計装着後 30 分と熟練度の低い状態で平均 80%以上の識別率が得られた．

識別結果に基づいて義手の指の角度制御を行う方法として，筋力の持続時間に基づいて指の目標角度を変える方法 Method 1 と，これに筋力の度合いも反映させて指の目標角度を変える方法 Method 2 の 2 種類を提案した．後者においては，動作に対応した筋の IEMG を用いる手法と，屈筋と伸筋の IEMG の差分を用いる手法の 2 つを提案した．

オンラインで筋電義手の指制御実験を行い，実験結果から，操作者の意思により義手の 2 指の角度を任意に制御できることを確認した．

謝辞

終始熱心なご指導を頂いた法政大学教授の石井千春教授に感謝の意を表します．

識別の実施及び制御実験にあたり，共同研究者の森田，植田，藤田をはじめとする研究室の面々にはひとかたならぬお世話になりました．多くの刺激と示唆を得ることができました．感謝の意を表します．ありがとうございました．最後に大学生活を続けていく過程でご支援いただいた家族，友人，本大学教授講師の方へ心から感謝の気持ちと御礼を申し上げたく，謝辞にかえさせていただきます．

参考文献

1) 田村宏樹, 奥村大, 淡野公一:「表面筋電位を FFT 処理しないで動作識別する方法の検討」, 電子情報通信学会論文誌.D, Vol.90, No.9, pp.2652-2655 (2007)

2) 齋藤怜，唐来拓，石井千春，佐々木智典，橋本洋志 :「階層型ニューラルネットワークと自己組織化マップによる筋電義手のための指動作識別の比較に関する研究」，産業計測制御研究会資料, pp.25-30 (2012)

3) 吉川雅博，三田友記，三河正彦，田中和世 :「前腕切断者を対象とした EMG に基づく動作識別に関する基礎研究」，人間工学, Vol.46, No.3, pp.197-207 (2010)

4) 原田昆寿・石井千春・疋田光孝 :「筋電義手のためのロボットハンドの設計と動作識別に関する研究」，計測自動制御学会 第 10 回システムインテグレーション部門講演会, pp.1899-1900 (2009)

5) 棒谷英法，大須賀美恵子 :「表面筋電信号に基づいた腕・手の動作識別」，人間工学, Vol.49, No.1, pp.1-9 (2013)

6) 末光厚夫 :「海馬型ニューラルネットワークに基づいた筋電義手制御のための動作推定システムの開発」，JAIST 学術研究成果リポジトリ b50. 科学研究費補助金研究成果報告書 (2011)

7) 井部鮎子，郷古学，伊藤宏司 :「表面筋電位を用いた前腕義手の複合動作識別」，計測自動制御学会論文集, Vol.45, No.12, pp.717-723 (2009)

8) 木曽淳，関弘和，南方英明，多田隈進 :「筋電義手制御を目的とした簡潔なニューラルネットワークによる前腕の動作識別」，ライフサポート, Vol.21, No.2, pp.55-62 (2009)

9) http://www.momo.cs.okayama-u.ac.jp/~takahashi/research.ja.html 岡山大学情報数理工学研究室

10) Information: Signals, Images, Systems: ISIS http://www.isis.ecs.soton.ac.uk/

付録

A) 特徴ベクトル作成 M 言語プログラミング

```
function [sys,x0,str,ts] = IEMG_save(t,x,u,flag)
switch flag,
  case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
  case 2,
    sys = mdlUpdate(t,x,u);
  case 3,
    sys = mdlOutputs(t,x,u);
  case 9,
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end

function [sys,x,str,ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 7;   %入力次元   7
sizes.NumOutputs     = 7;   %出力次元   7
sizes.NumInputs      = 7;   %入力次元   7
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x  = 0;
str = [];
ts  = [.01 0];              %動作周期   0.1s(10Hz)

function sys = mdlUpdate(t,x,u)
sys = u;%入力処理

function sys = mdlOutputs(t,x,u)
IEMG=x;
Tdata=x(5:6);
Count = load('Count.txt');%サンプル数加算, 連打防止
IEMG_max = load('IEMG_max.txt');%最大値の保存
IEMG_min = load('IEMG_min.txt');%最大値の保存
old_In=load('old_In.csv');%一つ前のIEMG
Samp_num=10;%各動作サンプル数
if(t>2)%IEMGのデータがそろうまで2待つ
    if(Tdata(1)==0&&Tdata(2)==0)%IEMG最大値の更新
        if(IEMG(1)>IEMG_max(1))
            IEMG_max(1)=IEMG(1);
        end
```

```
    if(IEMG(2)>IEMG_max(2))
        IEMG_max(2)=IEMG(2);
    end
    if(IEMG(3)>IEMG_max(3))
        IEMG_max(3)=IEMG(3);
    end
    if(IEMG(4)>IEMG_max(4))
        IEMG_max(4)=IEMG(4);
    end
    csvwrite('IEMG_max.txt',IEMG_max)
end

if(Tdata(1)==1&&Tdata(2)==0)%拇指学習
    if(IEMG(1)>(IEMG_max(1)*IEMG(7)))%入力がmaxのIEMG(7)を超えたら
        if(Samp_num>=Count(1)&Count(7)==0&IEMG(1)<(old_In(1)-.1))%カウント
            IEMG10(1, 1:4)=old_In(1:4);
            IEMG10(1, 5:6)=[1 0];%拇指一曲げ 指示ーNull
            csvwrite('IEMG10.txt',[load('IEMG10.txt');IEMG10])
            Count(1)=Count(1)+1;
            Count(7)=1;%連打防止
        end
    end

elseif(Tdata(1)==-1&&Tdata(2)==0)%拇指学習
    if(IEMG(2)>(IEMG_max(2)*IEMG(7)))%入力が指定上限値を超えたら
        if(Samp_num>=Count(2)&Count(7)==0&IEMG(2)<(old_In(2)-.1))%カウント
            IEMG_10(1, 1:4)=old_In(1:4);
            IEMG_10(1, 5:6)=[-1 0];%拇指一曲げ 指示ーNull
            csvwrite('IEMG-10.txt',[load('IEMG-10.txt');IEMG_10])
            Count(2)=Count(2)+1;
            Count(7)=1;%連打防止
        end
    end

elseif(Tdata(1)==0&&Tdata(2)==1)%示指学習
    if(IEMG(3)>(IEMG_max(3)*IEMG(7)))%入力がmaxのIEMG(7)を超えたら
        if(Samp_num>=Count(3)&Count(7)==0&(IEMG(3)<old_In(3)-.1))%カウント
            IEMG01(1, 1:4)=old_In(1:4);
            IEMG01(1, 5:6)=[0 1];%拇指一曲げ 指示ーNull
            csvwrite('IEMG01.txt',[load('IEMG01.txt');IEMG01])
            Count(3)=Count(3)+1;
            Count(7)=1;%連打防止
        end
    end

elseif(Tdata(1)==0&&Tdata(2)==-1)%示指学習
    if(IEMG(4)>(IEMG_max(4)*IEMG(7)))%入力が指定上限値を超えたら
        if(Samp_num>=Count(4)&Count(7)==0&(IEMG(4)<old_In(4)-.1))%カウント
            IEMG0_1(1, 1:4)=old_In(1:4);
```

```
                IEMGO_1(1, 5:6)=[0 -1];%拇指一曲げ 指示一Null
                csvwrite('IEMGO-1.txt',[load('IEMGO-1.txt');IEMGO_1])
                Count(4)=Count(4)+1;
                Count(7)=1;%連打防止
            end
        end

    elseif(Tdata(1)==1&&Tdata(2)==1)%両指学習
        if(Samp_num>=Count(5)&Count(7)==0)%入力がmaxのIEMG(7)を超えたら
            if((IEMG(1)>(IEMG_max(1)*IEMG(7)))&&(IEMG(1)<(old_In(1)-.1))||...
                (IEMG(3)>(IEMG_max(3)*IEMG(7)))&&(IEMG(3)<(old_In(3)-.1)))%カウント
                IEMG11(1, 1:4)=old_In(1:4);
                IEMG11(1, 5:6)=[1 1];%拇指一曲げ 指示一Null
                csvwrite('IEMG11.txt',[load('IEMG11.txt');IEMG11])
                Count(5)=Count(5)+1;
                Count(7)=1;%連打防止
            end
        end

    elseif(Tdata(1)==-1&&Tdata(2)==-1)%両指学習
        if(Samp_num>=Count(6)&Count(7)==0)%入力がmaxのIEMG(7)を超えたら
            if((IEMG(2)>(IEMG_max(2)*IEMG(7)))&&(IEMG(2)<(old_In(2)-.1))||...
                (IEMG(4)>(IEMG_max(4)*IEMG(7)))&&(IEMG(4)<(old_In(4)-.1)))%カウント
                IEMG_1_1(1, 1:4)=old_In(1:4);
                IEMG_1_1(1, 5:6)=[-1 -1];%拇指一曲げ 指示一Null
                csvwrite('IEMG-1-1.txt',[load('IEMG-1-1.txt');IEMG_1_1]);
                Count(6)=Count(6)+1;
                Count(7)=1;%連打防止
            end
        end
    end


if((IEMG(1)>(IEMG_max(1)*.2))&&(IEMG(2)>(IEMG_max(2)*.2))&&(IEMG(3)>(IEMG_max(3)*.2))
&&(IEMG(4)>(IEMG_max(4)*.2)))
            Count(7)=0;%基準値より下がったらパラメータが
end

csvwrite('old_In.csv',IEMG(1:4))%現在のデータを保存

    if(Tdata(1)==0&&Tdata(2)==0)%IEMG最大値の更新
        if(IEMG(1)<IEMG_min(1))
            IEMG_min(1)=IEMG(1);
            csvwrite('IEMG_min.txt',IEMG_min)
        end
        if(IEMG(2)<IEMG_min(2))
            IEMG_min(2)=IEMG(2);
            csvwrite('IEMG_min.txt',IEMG_min)
        end
```

```
        if(IEMG(3)<IEMG_min(3))
            IEMG_min(3)=IEMG(3);
            csvwrite('IEMG_min.txt',IEMG_min)
        end
        if(IEMG(4)<IEMG_min(4))
            IEMG_min(4)=IEMG(4);
            csvwrite('IEMG_min.txt',IEMG_min)
        end
    end

else%起動時初期化
    if(Tdata(1)==0&&Tdata(2)==0)
        csvwrite('IEMG_max.txt',[0,0,0,0])
        csvwrite('IEMG_min.txt',[10,10,10,10])
        csvwrite('IEMG11.txt',[]);
        csvwrite('IEMG01.txt',[]);
        csvwrite('IEMG10.txt',[]);
        csvwrite('IEMG-1-1.txt',[]);
        csvwrite('IEMG-10.txt',[]);
        csvwrite('IEMG0-1.txt',[]);
        Count=[1,1,1,1,1,1,0];
    end
    Count(7)=0;%連打防止
end

csvwrite('Count.txt',Count)

if(Tdata(1)==0&&Tdata(2)==0)
    sys = [IEMG_max 0 0 0];%出力処理
else
    sys = Count(1:7);%出力処理
end
```

B) SVM の abc 学習結果の保存ファンクション

```
function [ output_args ] = STU( input_args )
SVM_stua;
SVM_stub;
SVM_stuc;
IEMG_max=[0,load('IEMG_max.txt')];
IEMG_min=[0,load('IEMG_min.txt')];



function [ output_args ] = SVM_stua( input_args )
%   SVM-aの学習　開閉　曲げ1or反り-1
addpath 'svm';
Data=load('IEMG10.txt');
Data2=load('IEMG01.txt');
Data3=load('IEMG11.txt');
Data4=load('IEMG-10.txt');
Data5=load('IEMG0-1.txt');
Data6=load('IEMG-1-1.txt');

%学習用特長ベクトル X
X=[Data(:,1:4); Data2(:,1:4); Data3(:,1:4);
    Data4(:,1:4); Data5(:,1:4); Data6(:,1:4)];
Y=[Data(:,5); Data2(:,5); Data3(:,5);
    Data4(:,5); Data5(:,5); Data6(:,5)];
kernel = 'spline';

%Tdata作成
A = length(Y);
for i=1:A/2
   Y(i)=1;
end
for i=A/2+1:A
   Y(i)=-1;
end

[nsva, alphaa, b0a] = svc(X, Y, kernel);%SVM作成
 %学習データの保存
csvwrite('Xa.csv',X);
csvwrite('Ya.csv',Y);
csvwrite('alphaa.csv',alphaa);
csvwrite('ba.csv',b0a);



function [ output_args ] = SVM_stub( input_args )
%   SVM-bの学習　親指1　指-1
```

```
addpath 'svm';
Data=load('IEMG10.txt');
Data4=load('IEMG-10.txt');
Data2=load('IEMG01.txt');
Data5=load('IEMG0-1.txt');
X=[Data(:,1:4); Data4(:,1:4);
    Data5(:,1:4); Data2(:,1:4)];
Y=[Data(:,5); Data4(:,5);
    Data5(:,5); Data2(:,5)];
kernel = 'spline';
A = length(Y);
for i=1:A/2
   Y(i)=1;
end
for i=A/2+1:A
   Y(i)=-1;
end
[nsvb, alphab, b0b] = svc(X, Y, kernel);
csvwrite('Xb.csv',X);
csvwrite('Yb.csv',Y);
csvwrite('alphab.csv',alphab);
csvwrite('bb.csv',b0b);


function [ output_args ] = SVM_stuc( input_args )
%   SVM-cの学習　曲げ　全1 or else-1
addpath 'svm';
Data=load('IEMG10.txt');
Data2=load('IEMG01.txt');
Data3=load('IEMG11.txt');
Data4=load('IEMG-10.txt');
Data5=load('IEMG0-1.txt');
Data6=load('IEMG-1-1.txt');
X=[Data(:,1:4); Data2(:,1:4); Data3(:,1:4);
    Data4(:,1:4); Data5(:,1:4); Data6(:,1:4)];
Y1=[Data(:,5:6); Data2(:,5:6); Data3(:,5:6);
    Data4(:,5:6); Data5(:,5:6); Data6(:,5:6)];
Y=[Data(:,5); Data2(:,5); Data3(:,5);
    Data4(:,5); Data5(:,5); Data6(:,5)];
kernel = 'spline';
A = length(Y);
for i=1:A%SVM-c
   Y(i)=Y1(i, 1)*Y1(i, 2)*2-1;
end
[nsvc, alphac, b0c] = svc(X, Y, kernel);
csvwrite('Xc.csv',X);
csvwrite('Yc.csv',Y);
csvwrite('alphac.csv',alphac);
csvwrite('bc.csv',b0c);
```

C) 各 SVM-abc 識別ファンクション

```
function [sys,x0,str,ts] = SVM_c(t,x,u,flag)
switch flag,
  case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
  case 2,
    sys = mdlUpdate(t,x,u);
  case 3,
    sys = mdlOutputs(t,x,u);
  case 9,
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 5;
sizes.NumOutputs     = 1;   %出力次元   1
sizes.NumInputs      = 5;   %入力次元   5
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = 0;
str = [];
ts  = [.01 0];              %動作周期   0.01s(100Hz)
function sys = mdlUpdate(t,x,u)
sys = u;%入力処理
function sys = mdlOutputs(t,x,u)
tstX = x.';
addpath 'svm';
X=load('Xa.csv');
Y=load('Ya.csv');
alpha=load('alphaa.csv');
b0=load('ba.csv');
kernel = 'spline';
IEMG_max=load('IEMG_max.txt');
if(t<1)
    sys = [0];
else
    if(tstX(1)>IEMG_max(1)*tstX(5)||tstX(2)>IEMG_max(2)*tstX(5)...
            ||tstX(3)>IEMG_max(3)*tstX(5)||tstX(4)>IEMG_max(4)*tstX(5))
    tstOUTPUT = svcoutput(X, Y, tstX(1:4), kernel, alpha, b0);
    sys = tstOUTPUT;
    else
        sys = [0];
    end
```

```
end

function [sys,x0,str,ts] = SVM_b(t,x,u,flag)
switch flag,
  case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
  case 2,
    sys = mdlUpdate(t,x,u);
  case 3,
    sys = mdlOutputs(t,x,u);
  case 9,
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 5;
sizes.NumOutputs     = 1;   %出力次元   1
sizes.NumInputs      = 5;   %入力次元   5
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = 0;
str = [];
ts  = [.01 0];              %動作周期   0.01s(100Hz)
function sys = mdlUpdate(t,x,u)
sys = u;%入力処理
function sys = mdlOutputs(t,x,u)
tstX = x.';
addpath 'svm';
X=load('Xb.csv');
Y=load('Yb.csv');
alpha=load('alphab.csv');
b0=load('bb.csv');
kernel = 'spline';
IEMG_max=load('IEMG_max.txt');
if(t<1)
    sys = [0];
else
    if(tstX(1)>IEMG_max(1)*tstX(5)||tstX(2)>IEMG_max(2)*tstX(5)...
            ||tstX(3)>IEMG_max(3)*tstX(5)||tstX(4)>IEMG_max(4)*tstX(5))
    tstOUTPUT = svcoutput(X, Y, tstX(1:4), kernel, alpha, b0);
    sys = tstOUTPUT;
    else
        sys = [0];
    end
end
```

```
function [sys,x0,str,ts] = SVM_c(t,x,u,flag)
switch flag,
  case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
  case 2,
    sys = mdlUpdate(t,x,u);
  case 3,
    sys = mdlOutputs(t,x,u);
  case 9,
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 5;
sizes.NumOutputs     = 1;   %出力次元   1
sizes.NumInputs      = 5;   %入力次元   5
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = 0;
str = [];
ts  = [.01 0];              %動作周期   0.01s(100Hz)
function sys = mdlUpdate(t,x,u)
sys = u;%入力処理
function sys = mdlOutputs(t,x,u)
tstX = x.';
addpath 'svm';
X=load('Xc.csv');
Y=load('Yc.csv');
alpha=load('alphac.csv');
b0=load('bc.csv');
kernel = 'spline';
IEMG_max=load('IEMG_max.txt');
if(t<1)
    sys = [0];
else
    if(tstX(1)>IEMG_max(1)*tstX(5)||tstX(2)>IEMG_max(2)*tstX(5)...
            ||tstX(3)>IEMG_max(3)*tstX(5)||tstX(4)>IEMG_max(4)*tstX(5))
    tstOUTPUT = svcoutput(X, Y, tstX(1:4), kernel, alpha, b0);
    sys = tstOUTPUT;
    else
        sys = [0];
    end
end
```

D) SVM-abc 識別結果から識別結果を作成

```
function [sys,x0,str,ts] = ToMotor(t,x,u,flag)
switch flag,
  case 0,
    [sys,x0,str,ts] = mdlInitializeSizes;
  case 2,
    sys = mdlUpdate(t,x,u);
  case 3,
    sys = mdlOutputs(t,x,u);
  case 9,
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 8;
sizes.NumOutputs     = 2;   %出力次元   1
sizes.NumInputs      = 8;   %入力次元   1
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = 0;
str = [];
ts  = [.01 0];              %動作周期   0.1s(10Hz)
function sys = mdlUpdate(t,x,u)
sys = u;%入力処理
function sys = mdlOutputs(t,x,u)
tstX = x;%※行列注意
IEMG=x(4:7);
addpath 'svm';
Hand=load('Hand.csv');
IEMG_max = load('IEMG_max.txt');%最大値の保存
old_In=load('old_In.csv');%一つ前のIEMG

if(t<2)
    Hand=[0;0];
else
    if(Hand==0)%識別は最初のピーク時のみ
        if(((IEMG(1)>(IEMG_max(1)*tstX(8)))&&(IEMG(1)<(old_In(1)-.1)))||...
            ((IEMG(2)>(IEMG_max(2)*tstX(8)))&&(IEMG(2)<(old_In(2)-.1)))||...
            ((IEMG(3)>(IEMG_max(3)*tstX(8)))&&(IEMG(3)<(old_In(3)-.1)))||...
            ((IEMG(4)>(IEMG_max(4)*tstX(8)))&&(IEMG(4)<(old_In(4)-.1))))%
            if(tstX(1)==1)%SVM-a
                if(tstX(3)==1)%SVM-c
                    Hand=[1;1];
```

```
                elseif(tstX(2)==1)%SVM-b
                    Hand=[1;0];
                elseif(tstX(2)==-1)
                    Hand=[0;1];
                end
            elseif(tstX(1)==-1)%SVM-a
                if(tstX(3)==1)%SVM-c
                    Hand=[-1;-1];
                elseif(tstX(2)==1)%SVM-b
                    Hand=[-1;0];
                elseif(tstX(2)==-1)
                    Hand=[0;-1];
                end
            end
            csvwrite('Hand.csv',Hand);
        end
    elseif((Hand~=0)&&...
            ((IEMG(1)<(IEMG_max(1)*0.2))&&...
            (IEMG(2)<(IEMG_max(2)*0.2))&&...
            (IEMG(3)<(IEMG_max(3)*0.2))&&...
            (IEMG(4)<(IEMG_max(4)*0.2))))%IEMGが20%以下になっていたら初期化
        Hand=[0,0];
        csvwrite('Hand.csv',Hand);
    end
end

csvwrite('old_In.csv',IEMG(1:4))%現在のデータを保存
sys = Hand;
```